11장. 원시 값과 객체의 비교

11장. 원시 값과 객체의 비교

1. 자바스크립트가 제공하는 7가지 데이터 타입(숫자, 문자열, 불리언, null, undefined, 심벌, 객체)

   크게 **원시 타입(primitive type)**, **객체 타입(object/reference type)** 으로 구분

2. 원시 타입, 객체 타입의 차이점
   a. **원시 타입의 값, 즉 원시 값은 변경 불가능한 값**이다.

      이에 비해 **객체(참조) 타입의 값, 즉 객체는 변경 가능한 값**이다.

   b. **원시 값을 변수에 할당**하면 변수(확보된 메모리 공간)에는 **실제 값**이 저장.

      이에 비해 **객체를 변수에 할당하면** 변수(확보된 메모리 공간)에는 **참조 값**이 저장.

   c. **값에 의한 전달**

      : 원시 값을 갖는 변수를 다른 변수에 할당하면 원본의 **원시 값이 복사되어** 전달됨.

      참조에 의한 전달

      : 객체를 가리키는 변수를 다른 변수에 할당하면 원본의 **참조 값이 복사되어** 전달됨.

11.1 원시 값

11.1.1 변경 불가능한 값

1. **원시 타입의 값, 즉 원시 값은 변경 불가능한 값**이다.

   즉, 한번 생성된 원시 값은 읽기 전용(read only) 값으로서 변경할 수 없다.

2. **변경 불가능하다는 것은 변수가 아니라, 값에 대한 진술이다.**

   = 원시 값 자체를 변경할 수 없다는 것이지, 변수 값을 변경할 수 없다는 것이 아니다.

[예제 11 - 01]

```
// const 키워드를 사용해 선언한 변수는 재할당이 금지된다. 상수는 재할당
const o = {};

// const 키워드를 사용해 선언한 변수에 할당한 원시 값(상수)은 변경할
// 하지만, const 키워드를 사용해 선언한 변수에 할당한 객체는 변경할 수
o.a = 1;
console.log(o); // { a:1 }
```

3. 변수 값을 변경하기 위해 원시 값을 재할당하면 새로운 메모리 공간을 확보하고 재할당한 값을

   저장한 후, 변수가 참조하던 메모리 공간의 주소를 변경.

   ➡️ **불변성(immutability)**

   (불변성을 갖는 원시 값을 할당한 변수는, 재할당 이외에 변수 값을 변경할 수 있는 방법이 없다.)

11.1.2 문자열과 불변성

1. 문자열은 변경 불가능한 값이다.

2. 한번 생성된 문자열은 읽기 전용 값으로서 변경할 수 없다.

11.1.3 값에 의한 전달

1. **값에 의한 전달**

   : 변수에 원시 값을 갖는 변수를 할당하면 할당받는 변수(copy)에는

   할당되는 변수(score)의 원시 값이 복사되어 전달된다.

2 . **“값에 의한 전달”**도 사실은 값을 전달하는 것이 아니라, **메모리 주소를 전달**한다.

단, **전달된 메모리 주소를 통해 메모리 공간에 접근하면 값을 참조할 수 있다.**

11.2 객체

11.2.1 변경 가능한 값

1. **객체(참조) 타입의 값, 즉 객체는 변경 가능한 값(mutable value)**이다.

2. 객체를 할당한 변수는 재할당 없이 객체를 직접 변경할 수 있다.

   즉, 재할당 없이 프로퍼티를 동적으로 추가할 수도 있고, 프로퍼티 값을 갱신할 수도 있으며

   프로퍼티 자체를 삭제할 수도 있다.

   [예제 11 - 13]

```
var person = {
    name : "Lee"
};

// 프로퍼티 값 갱신
person.name = "Kim";

// 프로퍼티 동적 생성
person.address = "Seoul";

console.log(person); // { name:"Kim", address:"Seoul" }
```

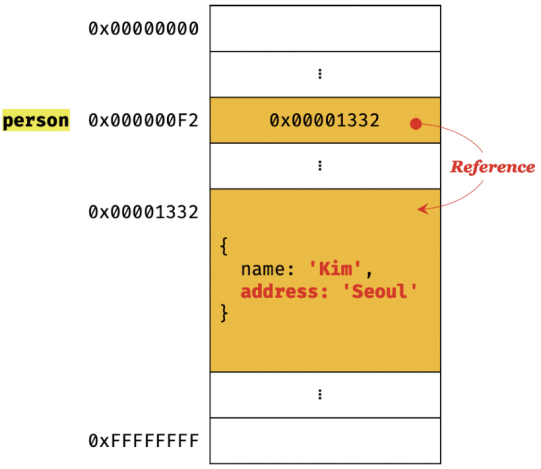

그림 11-8 객체는 변경 가능한 값이다.

- 객체를 생성 & 관리하는 방식은 매우 복잡하고 비용이 많이 드는 일

3. 객체의 이러한 **구조적 단점에 따른 부작용**

   → 원시 값과는 다르게, **여러개의 식별자가 하나의 객체를 공유할 수 있다**는 것.

11.2.2 참조에 의한 전달

1. [예제 11-16]

```
var perosn = {
    name : 'Lee'
};

// 참조 값을 복사(얕은 복사)
var copy = person;
```

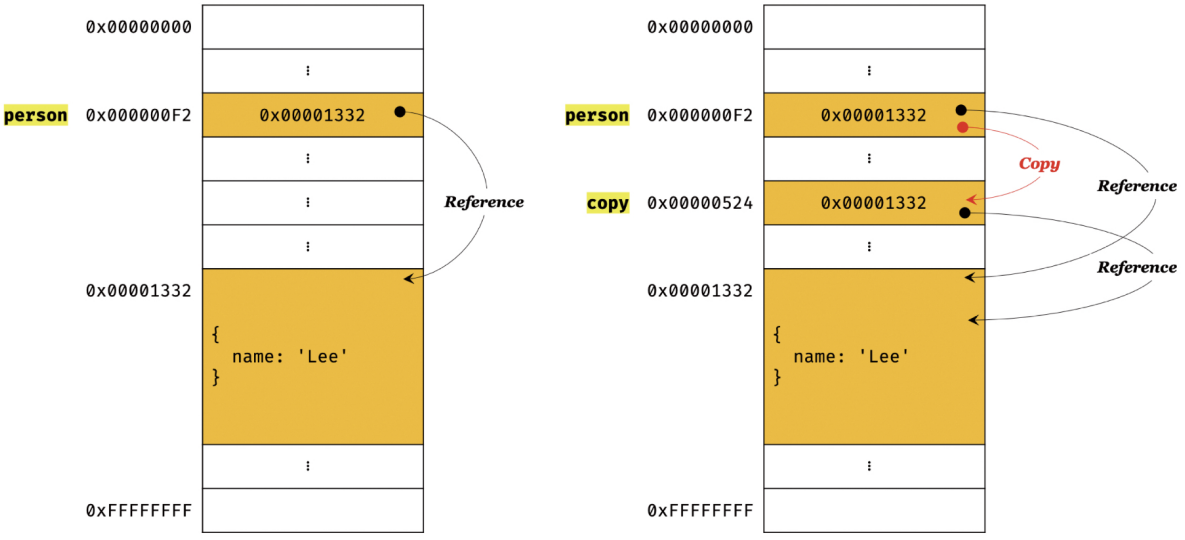

그림 11-9 참조에 의한 전달

- 원본 person, 사본 copy - 저장된 메모리 주소는 다르지만, 동일한 참조 값을 갖는다.
- **두 개의 식별자가 하나의 객체를 공유**한다는 것을 의미

2. [예제 11-18]

```
var person1 = {
    name : 'Lee'
};

var person2 = {
    name : 'Lee'
};

console.log(person1 === person2);  // 1번 - False
console.log(person1.name === person2.name);  // 2번 - True
```